Random Forest

Ariel Romero

library(tidyverse)

## -- Attaching packages -------------

## v ggplot2 3.1.0 v purrr 0.3.2   
## v tibble 2.1.1 v dplyr 0.8.0.1  
## v tidyr 0.8.3 v stringr 1.4.0   
## v readr 1.3.1 v forcats 0.4.0

## -- Conflicts ----------------------  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library(ranger)

Blood <- read\_csv("Blood.csv")

## Parsed with column specification:  
## cols(  
## Mnths\_Since\_Last = col\_double(),  
## TotalDonations = col\_double(),  
## Total\_Donated = col\_double(),  
## Mnths\_Since\_First = col\_double(),  
## DonatedMarch = col\_double()  
## )

Blood = Blood %>% mutate(DonatedMarch=as\_factor(as.character(DonatedMarch))) %>% mutate(DonatedMarch = fct\_recode(DonatedMarch, "No" = "0", "Yes" = "1"))  
  
str(Blood)

## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 748 obs. of 5 variables:  
## $ Mnths\_Since\_Last : num 2 0 1 2 1 4 2 1 2 5 ...  
## $ TotalDonations : num 50 13 16 20 24 4 7 12 9 46 ...  
## $ Total\_Donated : num 12500 3250 4000 5000 6000 1000 1750 3000 2250 11500 ...  
## $ Mnths\_Since\_First: num 98 28 35 45 77 4 14 35 22 98 ...  
## $ DonatedMarch : Factor w/ 2 levels "Yes","No": 1 1 1 1 2 2 1 2 1 1 ...

set.seed(1234)  
train.rows = createDataPartition(y = Blood$DonatedMarch, p=0.7, list = FALSE)  
train = Blood[train.rows,]   
test = Blood[-train.rows,]

fit\_control = trainControl(method = "cv",   
 number = 10)   
  
set.seed(123)   
rf\_fit = train(x=as.matrix(Blood[,-5]), y=as.matrix(Blood$DonatedMarch),  
 method = "ranger",   
 importance = "permutation",  
 trControl = fit\_control,num.trees = 100)

varImp(rf\_fit)

## ranger variable importance  
##   
## Overall  
## TotalDonations 100.00  
## Mnths\_Since\_First 78.15  
## Mnths\_Since\_Last 18.51  
## Total\_Donated 0.00

rf\_fit

## Random Forest   
##   
## 748 samples  
## 4 predictor  
## 2 classes: 'No', 'Yes'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 673, 673, 673, 673, 673, 673, ...   
## Resampling results across tuning parameters:  
##   
## mtry splitrule Accuracy Kappa   
## 2 gini 0.7674054 0.2638705  
## 2 extratrees 0.7860721 0.2920425  
## 3 gini 0.7527027 0.2242451  
## 3 extratrees 0.7594234 0.2418346  
## 4 gini 0.7460721 0.2171814  
## 4 extratrees 0.7460721 0.2140371  
##   
## Tuning parameter 'min.node.size' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were mtry = 2, splitrule =  
## extratrees and min.node.size = 1.

TotalDonation is the most important variable in the mode, Total\_Donated is the least important.

predRF = predict(rf\_fit, train)  
head(predRF)

## [1] Yes Yes No No Yes Yes  
## Levels: No Yes

confusionMatrix(predRF,train$DonatedMarch)

## Warning in confusionMatrix.default(predRF, train$DonatedMarch): Levels are  
## not in the same order for reference and data. Refactoring data to match.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 74 3  
## No 51 396  
##   
## Accuracy : 0.8969   
## 95% CI : (0.8677, 0.9216)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 1.614e-15   
##   
## Kappa : 0.6732   
##   
## Mcnemar's Test P-Value : 1.596e-10   
##   
## Sensitivity : 0.5920   
## Specificity : 0.9925   
## Pos Pred Value : 0.9610   
## Neg Pred Value : 0.8859   
## Prevalence : 0.2385   
## Detection Rate : 0.1412   
## Detection Prevalence : 0.1469   
## Balanced Accuracy : 0.7922   
##   
## 'Positive' Class : Yes   
##

The accuracy is 89.69%, senesitiyvy is 58.40% and specificity is 99.50%.

The accuracy is statistically signficant better than the naive.

predrf2 = predict(rf\_fit,test)  
  
head(predrf2)

## [1] Yes Yes No No No Yes  
## Levels: No Yes

confusionMatrix(predrf2, test$DonatedMarch)

## Warning in confusionMatrix.default(predrf2, test$DonatedMarch): Levels are  
## not in the same order for reference and data. Refactoring data to match.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 26 3  
## No 27 168  
##   
## Accuracy : 0.8661   
## 95% CI : (0.8144, 0.9078)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 9.092e-05   
##   
## Kappa : 0.5606   
##   
## Mcnemar's Test P-Value : 2.679e-05   
##   
## Sensitivity : 0.4906   
## Specificity : 0.9825   
## Pos Pred Value : 0.8966   
## Neg Pred Value : 0.8615   
## Prevalence : 0.2366   
## Detection Rate : 0.1161   
## Detection Prevalence : 0.1295   
## Balanced Accuracy : 0.7365   
##   
## 'Positive' Class : Yes   
##

The model performs almost exactly like it did for the training set.

In the real world, this model could be used to predict who will be most likely to donate in March. This could prove useful in a case where there is a severe problem shortage to get a count of how many would donate blood. I would recommend this model for a high level prediction but for more granular information, there would have to more data provided in the dataset.